The Search for an Empirical and Theoretical Foundation for Algorithm Visualization

Christopher D. Hundhausen
Department of Computer and Information Science
University of Oregon
Eugene, OR 97403-1202
chundhau@cs.uoregon.edu

Table of Contents

The Search for an Empirical and Theoretical Foundation for Algorithm Visualization

> One's object is then to have a clear mental picture of the state of the machine at each moment in the computation. This object can only be achieved with a struggle.
>
> (Turing 1950, p. 459)

1 Introduction

Scientists have always resorted to illustrations, figures and diagrams to elucidate the difficult concepts they are trying to explain and understand, and computer scientists are no exception. With the advent of powerful graphical workstations in the 1980s, illuminating diagrams in computer science moved from paper to computer screen, as a new area of computer science—algorithm visualization (AV)—emerged as a way to perceive computer programs in execution. Beginning with the development of Marc Brown's BALSA system (Brown, 1988) during the 1980s, several graphical software systems were developed that facilitated the viewing of a computer algorithm through a series of static pictures (e.g. Naps & Hundhausen, 1991) or through animated movies (e.g. Brown, 1988; Stasko, 1989).

In the introduction of his seminal dissertation *Algorithm Animation* (1988), Brown makes two general and rather profound statements about algorithm animation environments, the ramifications of which he never actually examines. First, he claims that they make "possible a fundamental improvement in the way we understand and think about [algorithms]" (p. 1). And second, he claims that they expose "properties of [a computer] program that might otherwise be difficult to understand or might even remain unnoticed" (p. 1). At first glance, these claims might seem obvious, intuitive and innocuous. Yet, as research in algorithm animation and visualization continues to be enthusiastically propagated, I have become increasingly wary of a discipline whose porous theoretical foundation is rooted primarily in the intuition of its practitioners. Futernick (1989) well identifies the problem in his recent dissertation: "Recently, computers have been used to teach computer programming . . . In spite of the enormous sums of money being channeled into these computer applications, little effort has been made among educational theorists to identify and evaluate the theoretical assumptions supporting them." Indeed, it seems that computer scientists are quick to make claims and assumptions about the pedagogical value of animation and visualization systems, but are derelict when it comes to corroborating those claims and assumptions with theories or empirical research.

Although one could argue that such theories and empirical research "are best left to the psychologists"[1], I believe that, as computer scientists, we have no business touting visualization systems as "effective" and "educationally beneficial" without qualifying or scrutinizing our claims theoretically and empirically. In this article, we shall first introduce algorithm visualization, and describe one algorithm visualization system (Naps & Hundhausen, 1991) in particular. We shall then identify the (tacit) assumptions that have undergirded its development and advancement, and subject those assumptions to the scrutiny of the research and theories of cognitive science. Finally, we shall suggest some ways in which algorithm visualization might be more effectively utilized, and we shall identify some directions for future cognitive and computer research in this area.

[1] My undergraduate adviser often reassured me with these words.

2 Algorithm Visualization: Background and Overview

In this section, we first define algorithm visualization and outline its goals. Next,we present an example to illustrate to how it might be used. We then outline the specifics of GAIGS (Naps & Hundhausen, 1991), the specific system we wish to scrutinize: the techniques used to build visualizations, and the specific environments in which students have interacted with them.

2.1 Definition

Recognizing that "algorithms in action are complex objects whose properties can be difficult to fathom (Brown 1984, 177), algorithm visualization (AV) purports to provide a graphically oriented method for teaching and studying algorithms. A good general definition of algorithm visualization is as follows:

> **Algorithm visualization** — the process of viewing the underlying logic of a computer algorithm through a series of pictures that are strategically chosen to illustrate the algorithm in execution.

Implicit to this definition are three general goals. First, algorithm visualization attempts to associate the set of actions taken by an algorithm with a corresponding set of illustrations. These illustrations generally take the form of snapshots of the manipulated data structure(s) at crucial points in the algorithm. Second, realizing that the visual sense often contributes greatly to human learning, algorithm visualization tries—by associating actions with illustrations—to accelerate and enhance the process of learning the underlying logic of computer algorithms. Third, whereas traditional teaching methods have suggested an inextricable relationship between an algorithm and its implementation in a programming language, algorithm visualization strives to separate and distinguish the algorithm from its implementation. Because algorithms in an algorithm visualization system are depicted as a series of pictures, no specific programming language need be employed to define the algorithm; it can be learned and understood as an abstract, but visually identifiable, sequence of actions.

2.2 An Example

To give some substance to this definition, let us examine an illustrative and frequently used application of AV: sorting algorithms. Given a list of *n* integers, a sorting algorithm should arrange the numbers into some specified order—either ascending or descending. The integers to be sorted are assumed to reside in a data structure called an *array*, which is nothing more than a list whose elements are randomly accessible. A sorting algorithm, then, must make a series of data comparisons and swaps that place the integers in order. For example, if the sorting algorithm's task were to sort the integers into ascending order, then, upon completion of the sort, the smallest integer would reside in the first array cell, the second smallest integer would reside in the second array cell, and so on.

Because of its simplicity, *selection sort* is often used to introduce sorting to beginning computer scientists. One popular introductory book on the Pascal programming language (Cooper & Clancy, 1985) presents selection sort as follows:

> In a selection sort, we find the smallest array element and exchange it with the array's first element, then find the second smallest element and exchange it with the array's second element, etc. (p. 529).

After a loose English definition, it is appropriate to formalize the algorithm a bit with *pseudocode*, a notation more formal than English, but less formal than a computer language:

> ***for*** *every 'first' element in the array*
> *find the largest element in the array;*
> *exchange it with the 'first' element;* (ibid., p. 528)

Notice that in this pseudocode, the symbol *'first'* acts implicitly as a variable that takes on the value of each array index in turn; the multiple values of 'first' are facilitated by the *for every* that precedes it. The notions of variables and loops are made explicit in a programming language version of selection sort, which culminates the formalization. In Pascal, selection sort appears as follows (ibid., p. 528):

```
procedure SelectionSort (var Data: TheArrayType);
     {Sorts array Data using selection sort.}
     var First, Current, Least:  integer;
     begin
        for First := 1 to ARRAYLIMIT – 1 do begin
           Least := First; {Take a guess that this is the least value.}
           for Current := First + 1 to ARRAYLIMIT do
              if Data[Current] < Data[Least] then Least := Current;
              {Look for smaller value in the remainder of the array.}
              Swap(Data[Least], Data[First]){Assume this procedure exchanges
                                             the elements.}
        end {outer for}
     end; {SelectionSort}
```

In contrast to these textual descriptions of selection sort above, the AV technique (Hundhausen & Naps, 1991) we shall describe in more detail below presents selection sort as a series of "snapshots" of the array being sorted. Assuming an initial array containing five integers—4,2,5,3,1—a visualization of the selection sort algorithm might appear as in Figure 1. Notice that the visualization comprises nothing other than a picture of the array after each iteration of selection sort's main (outer) loop. Although the array could have been pictured more frequently—say, after each data swap—the essential logic of the algorithm can be seen with only these six snapshots.

(Figure omitted.)

Figure 1. A GAIGS visualization of selection sort on an array of five data elements. The array is pictured after each iteration of the main loop, and the caption above each array snapshot provides a running total of the number of total loop iterations (inner + outer), the number of data swaps that have taken place, and the number of comparisons between two data elements that have taken place.

2.3 The GAIGS AV System

2.3.1 Origins and technique

GAIGS (Generalized Algorithm Illustration through Graphical Software) was developed at Lawrence University under an NSF ILI grant during the late 1980s and early 1990s; it has been used extensively in Lawrence's undergraduate computer science curriculum since 1988. The original conception of GAIGS (Naps, 1988) was motivated by a recognition that the manipulation of one or more data structures plays a fundamental role in—indeed, is the essence of—the execution of computer algorithms. A natural way to visualize an algorithm, therefore, is to view a series of data structure snapshots—appropriately narrated—that depict the effect of the algorithm on the data structure(s). And this is exactly the way one creates algorithm visualizations with GAIGS: one captures in a text file a series of data structure snapshots that result from the execution of the

algorithm, and replays the text file in the GAIGS graphical environment, in which the data structure snapshots take pictorial form. See Figure 2 for an example of a visualization within the GAIGS environment.

(Figure omitted.)

Figure 2. A sample GAIGS environment screen. This particular visualization shows five alternative, concurrent views (array, array by relative magnitude, array by scatterplot, stack of recursive calls, and calling tree) of Quicksort, a popular sorting algorithm.

2.3.2 The interesting events paradigm

As mentioned above, visualizations are created during the execution of an algorithm via snapshots of the instantaneous states of the algorithm's data structure(s). Although the details of the exact method (in computer code) by which one takes a data structure snapshot within an algorithm are not salient to the purposes of this paper[2], it is important to ask how one chooses the appropriate points in an algorithm at which to take a snapshot of a data structure.

The fundamental operations of an algorithm are what Brown (1988) labeled *interesting events*:: "interesting phenomena that should give rise to some type of display" (p. 12). Hence, the choice reduces to deciding which operations in an algorithm are fundamental to its execution—that is, which operations capture the essence of the logic behind the algorithm. In general, this question is nontrivial, and is best left to the algorithm author or some expert familiar with the algorithm and experienced in visualization.

Notice that the choice of interesting events could potentially effect the way in which the user understands the algorithm, since a choice of one set of interesting events over another implicitly emphasizes the importance of some operations and de-emphasizes the importance of others. As it turns out, however, experts familiar with the algorithm generally agree on which points in the algorithm give rise to interesting events. In fact, the interesting question is not which events are interesting, but the level of detail to include in the visualization., the most important factor governing interesting event selection is the skill level of the intended viewers of the visualization: the more advanced the viewer, the less detail is needed for the visualization to be meaningful.

2.3.3 Contexts in which GAIGS has been used

GAIGS has been used at Lawrence in two distinct settings: as a teaching aid in the classroom, and as the interactive software for structured exercises in the laboratory. It is necessary to describe the nature of each of these uses in a bit more detail, to give some context to our discussions

Classroom. Recall the selection sort example presented above. The way in which that example was presented well illustrates the way in which teachers typically present new algorithms in the classroom: they start with the its abstract form (e.g. a high level description of what sorting does), and progress toward its most concrete form (e.g., a visualization of one particular sorting algorithm (selection sort) working on the data set 5,3,2,1,4). Hence, by the time GAIGS is employed in the classroom, it is assumed that students have obtained sufficient declarative knowledge—whether through a textbook or through an introduction in the lecture—to interpret the visualization meaningfully. Although the teacher is the one interactively manipulating and narrating a visualization, students will often be asked to intervene with conjectures as to what the next data

[2] See (Hundhausen, 1990) for complete details.

structure picture should look like. In addition, a teacher might react to particular questions raised in class—for example, "how does selection sort handle data already in order—by presenting a relevant visualization example that answers the question. As can be seen, in the classroom GAIGS serves both as an medium in which to present concrete examples of algorithms already discussed at higher levels of abstraction, and as a tool for facilitating and stimulating class participation.

Laboratory. In the laboratory, GAIGS serves as the interactive environment for structured exercises. At Lawrence, groups of ten students (two to a machine) are led by a student or faculty lab assistant in a two hour weekly lab session in conjunction with a computer science course. The lab exercise have at least three goals: (1) to facilitate the interactive exploration of the week's principles; (2) to reinforce the declarative knowledge learned in class; and (3) to address particular issues, or to explore nuances of algorithms, that cannot be specifically addressed in the lectures. A typical lab exercise, such as the one presented in Figure 3, might be to modify an algorithm presented during the week's lectures, and then to use GAIGS to compare the modified algorithm to the original.

Shell Sort Lab

Each of the questions/exercises below requires a written response. Those responses are to be developed on your own after the lab period, making use of your observations during the lab session.

1. The worst case for insertion sort on a 8 element array is demonstrated by the following list of values:

   80 70 60 50 40 30 20 10

   To make sure that you understand how shell sort would manipulate this array, trace the action of shell sort on this array by running the program LAB1A and observing the visualization that results. How many comparisons are required by shell sort on this array? How does this number of comparisons compare to the number that would be required for straight insertion sort on the same array? Verify this by running the SHELLA program and observing the results of racing insertion sort against shell sort.

2. Race shell sort against insertion sort for some larger arrays. Observe the visualizations which result from these races. Summarize the results of your observations in short essay form. In this essay, you should be able to derive some specific conclusions about the run-time efficiency of insertion sort versus shell sort. Back up these conclusions with the results you observe in the races.

3. Identify a problem with the sequence of diminishing increments which you feel should perform better than the sequence presently in shell sort.

4. Specify at least two other sequences of diminishing increments which you feel should perform better than the sequence presently in shell sort.

5. Implement the first of the sequences you specified for 4. Verify that you have implemented it correctly by using SHELLA to trace it on a small array. Then, using SHELLB, race your new sort against the original shell sort for a variety of array data. (Implement your new sort by making changes to the UserDefinedSort PROCEDURE in the SHELLB file.) Summarize the results of your observations in short essay form.

6. Repeat 5 for the second of the sequences specified in 4.

Figure 3. A typical GAIGS laboratory exercise

3 Assumptions Underlying Development and Advancement of AV

In this section, we explicate the tacit assumptions that have undergirded the development and advancement of the AV technique described above. As I see it, three basic assumptions have been made. Beginning with the most general, and moving to the most AV specific, we shall examine each assumption and identify some problems associated with it.

3.1 General: Picture better than words

Description. AV is just one instance in which the widely accepted proverb "A picture is worth 10,000 words" has been taken to heart. Relying mainly on their intuition that the visual sense is powerful, computer scientists have propagated a variety of visualization techniques on the premise that they lead to an enhanced, deeper understanding of the processes being studied; that they reveal properties of the phenomena being studied that might otherwise go unnoticed; and that they facilitate faster and enhanced learning of the concepts being studied.

Problems. The belief that a picture is better than words needs to be qualified more precisely; we need to ask what "better" means; we need to identify specifically the ways in which pictures are "better" than words; and we need to determine whether the empirical research supporting the assumption applies to the contexts in which AV is used.

3.2 Computer Science Specific: Graphical software motivates learning

Description. Captivated by their "sexy" graphics, and by the novel ways in which they allow one to perceive a process, computer scientists have advanced visualization systems on the merit of their aesthetically pleasing appearance.

Problems. A sexy, novel, and interactive graphical software system can be captivating, but that does not necessarily mean that it has contributes to learning. We need to identify more specifically how sexy, novel, and interactive software might motivate learning.

3.3 AV Specific: AV facilitates the learning and understanding of algorithms better than conventional textually-based methods.

Description. This assumption lies at the heart of the GAIGS project described earlier; indeed, in our recent paper (Naps & Hundhausen, 1991), we even made the claim explicit: "algorithm visualization. . .has emerged as a more effective. . .technique for teaching, studying, and analyzing [algorithms]" (p. 259). Although we did not clarify the ways in which AV was "more effective" than traditional practices, we did assume that associating actions with illustrations might accelerate and enhance the process of learning the underlying logic of computer algorithms.

Problems. Once again, one should be wary of broad and imprecise claims about a system that are not accompanied by proof; clearly, specific empirical evidence is needed to support such claims. As can be seen, imprecise and unsupported assumptions run ubiquitously throughout AV. A reformulation of those assumptions, along with a search for corroborating evidence, is in order.

4 A cognitive science analysis of the assumptions underlying AV

In this section, we consolidate the cognitive science research salient to analyzing the three assumptions of algorithm visualization outlined above. Each subsection begins by formulating the assumption more precisely—in terms that better lend themselves to scientific analysis. It then presents the relevant cognitive science research, and ends with an introspective discussion. As we shall learn, although ostensibly innocuous on the surface, these assumptions have undergone sophisticated analysis in the realm of cognitive science. Moreover, unless we are willing to qualify and focus considerably our qualitative assumptions about algorithm visualization, we find that empirical studies really do not say much to corroborate them.

4.1 A picture is better than text

4.1.1 Reformulation

As discussed above, the comparative "better than" means very little unless qualified. To qualify it in the present case, we need to ask ourselves how a picture might be "better than" text in the learning of an algorithm. While we will save an analysis of the value of pictures versus text in the specific contexts of learning computer programming for Section 4.3 below—in which the specific pedagogical assumption underlying AV will be examined—we can make at least two broad qualitative statements about what it means for a picture to be "better than" words, both which certainly could apply to the learning of an algorithm.

First, if pictures are truly better, we should be able to remember them more easily and readily than we remember words. That is, given a picture and a textual description that are semantically equivalent representations of the same concept, the picture should facilitate recall of the concept better than the textual description. If pictorial recall superiority holds true, then the act of viewing algorithms through pictures that are semantically equivalent to conventionally used words would afford students a better memory of the concepts they were studying. Second, the Chinese put their fingers on another way in which pictures can be better than words when they offered the world their ancient proverb "A picture is worth 10,000 words." Indeed, it well connotes the generally held belief that pictures provide a more efficient encoding of concepts than do words. In particular, the information that we get "for free" in a picture (i.e. we can infer it) would take a great number of words (around 10,000) to express explicitly. In the following section, we shall inspect some of the empirical evidence and cognitive theories that either support the superiority of pictures to words, or identify the specific conditions under which pictures are superior.

4.1.2 Theories and empirical evidence

Origins. The earliest known experimental study of humans' ability to recall pictures and words was undertaken by Kirkpatrick in 1894. He presented lists of items—both pictures and words—to 329 male and female grade school through college students. The subjects were asked to recall the objects at two different times: immediately, and seventy two hours later. Kirkpatrick's results, presented in Figure 4, indicate a remarkable difference between the human recall rates of pictures and words. Curiously, although Kirkpatrick's results were replicated just four years later by Calkins (Madigan 1983, p. 65), further research on pictorial versus verbal memory remained sparse until the mid 1960s, owing to the decline of cognitive theory in general during the first half of this century (ibid., p. 66).

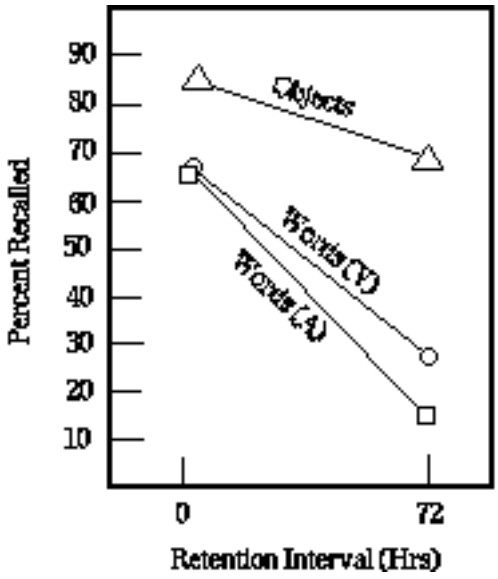

Figure 4. Summary graph of Kirkpatrick's seminal tests. The three lines depict subjects' retention levels of objects, auditory (A) words, and visual (V) words over a 72 hour period. (Adapted from Madigan, 1983, p. 66).

Contemporary research. In the 1960s and early 1970s, large scale empirical studies such as (Nickerson, 1965; Shepard, 1967; Haber, 1970; Standing, 1973) reinforced the differences in picture and verbal recall that were originally observed in the seminal experiments described above. For example, in Standing's study, subjects were presented anywhere from ten to 10,000 complex pictures, and were able to recall subsets of them with 95% accuracy. In a similar vein, Shepard (1967) tested the ability of 612 subjects to recognize pictures at various intervals ranging from two hours to 120 days after first seeing them. He found that they could still recognize the pictures with an accuracy of 92% after a week, with recognition accuracy fading to 57% after 120 days.

With plentiful empirical studies in hand, cognitive psychologists moved in two different directions during the 1970s and 1980s to explain the differences observed in picture and text comprehension and recall. One group of researchers (e.g. Paivio, 1971; Nelson, 1979) formulated theories on the *internal representation* of pictures and words in the brain to explain the differences; another group (e.g. Anderson, 1984; Larkin & Simon, 1987) devised theories predicated on the idea that the *cognitive processes* (operators) available to apply to the internal representation are more important than the representation itself. Beginning with the internal representation theories, and moving to the process-oriented theories, we examine below two representative examples of each line of research.

Paivio's dual coding theory. The main assumption underlying the dual coding theory (Paivio, 1971) is that "cognition consists largely of the activity of two partly interconnected but functionally independent and distinct symbolic systems" (Paivio 1983, p. 308). One encodes verbal events (words); the other encodes nonverbal events (pictures). By *interconnected*, Paivio means "that representations in one system can activate those in the other, so that, for example, pictures can be named and images can occur to words" (Madigan 1983, p. 80). *Functionally independent*, according to Paivio, implies that "nonverbal (imaginal) and verbal memory codes, aroused directly by pictures and words or indirectly by imagery and verbal encoding tasks, should have additive effects on recall" (ibid., p. 80). Therefore, assuming that pictures are more likely to be dually encoded than

words—which Paivio does (Kobayashi 1986, p. 784)—we should be able to remember pictures better than words, since there exist redundant, additive encodings of them.

In (Paivio 1975) and (Paivio & Csapo 1973), the authors offer a rather convincing empirical case for the dual-coding of pictures. They show that "presentation of a pictorial representation of an item and presentation of a verbal representation of the same concept have independent and additive effects on recall, unlike picture-picture or word-word representations" (Madigan 1983, p. 80). This is just one example of over sixty empirical studies in support of dual coding that Paivio has documented (Paivio, 1983). Although the range of these studies is broad— from imagery concreteness effects of language to modality-specific interference to neuropsychological evidence—three general features are shared by all. First, all studies can be accounted for by the same, relatively small set of theoretical assumptions. Second, the studies confirm the theory's ability to "predict specific changes and even reversals of effects as a function of task conditions" (ibid., p. 313). And third, the studies reveal that "different classes of variables often produce parallel effects, presumably because the variables converge on the same underlying process" (ibid., p. 313-314).

Nelson's sensory-semantic model. The sensory-semantic model (Nelson, 1979) is based on three main assumptions. First, it assumes that "both pictures and words access a common semantic code" (Kobayashi 1986, p. 785); moreover, it assumes that pictures access this information more directly than do words. Second, whereas "pictures access phonemic information about their verbal labels after semantic access," it is assumed that "words must progress through access to phonemic information prior to activation of semantic codes." And third, the model assumes that pictures and words have different sensory codes, which stem both from "the differences in the sensory and physical features of pictures and words," and from the fact that pictures are more discriminable and more distinctive than words" (Kobayashi 1986, 786).

An example of a study that lends credence to Nelson's model can be found in (Nelson, Reed, & Walling, 1976). They predicted that if they showed subjects a set of pictures of objects with similar shape, but with low conceptual similarity—for example, a knife, a bat, and a nail—pictures would no longer be better cues than words. As predicted, the authors found that "pictures were no better as cues than words at a relatively slow rate of presentation (2.1 sec), and [that they] were actually less efficient cues at a faster rate of presentation (1.1 sec)" (Kobayashi 1986, p. 786).

Anderson's tri-code theory. Whereas Paivio's theory assumed two types of mental encoding, Anderson (1983) promulgates a theory with "three codes or representation types: a temporal string, which encodes the order of a set of items; a spatial image, which encodes spatial configuration; and an abstract proposition, which encodes meaning" (Anderson 1983, p. 45). For readers familiar with Anderson's earlier work (e.g. Anderson, 1978), in which he maintained that it is impossible to discriminate among various representational notations, his tri-code theory might seem contradictory. However, Anderson is careful to point out that his most recent advocacy of a tri-code theory does not address the notation used to express the encodings, but rather, the processes that are defined on those encodings. Whereas he claims "it is impossible to identify whether a particular notation correctly expresses the structure of a representation or whether different knowledge structures are encoded according to different notations, [he claims]. . . it is possible to decide that different knowledge structures have different processes defined upon them" (Anderson 1983, p. 46). Hence, Anderson believes that the mental representation of pictures and words can only be determined by the processes that act on them. He argues elsewhere (Anderson, 1987) that the value of a certain representation will always depend on issues of processing efficiency, and not on the representation itself.

Larkin & Simon's analysis of sentential and diagrammatic representations. Unlike the researchers discussed above, Larkin & Simon (1987) completely circumvented the question of whether the *internal representations* of pictures and text can be distinguished. Instead, they were

interested in the computational differences to which the differing *external representations* of diagrams and sentences might give rise.

In order to conduct such an analysis, they first had to define quite precisely what they meant by a *sentential* and a *diagrammatic representation*: Whereas "in a *sentential* representation, the expressions form a sequence corresponding, on a one-to-one basis, to the sentences in a natural-language description of the problem,. . .in a *diagrammatic* representation, the expressions correspond, on a one-to-one basis, to the components of a diagram describing the problem" (Larkin & Simon 1987, p. 66). With these definitions in place, Larkin & Simon relied on the concepts of *informational* and *computational equivalence* to compare representations: "Two representations are *informationally equivalent* if all of the information in the one is also inferable from the other, and vice versa. . .[, and]. . .two representations are *computationally equivalent* if they are informationally equivalent, and, in addition, any inference that can be drawn easily and quickly from the information given explicitly in the one can also be drawn easily and quickly from the information given explicitly in the other, and vice versa" (ibid., p. 67). Given that a diagrammatic representation and a sentential representation are *informationally equivalent*, the *better* representation, according to Larkin & Simon, is the one that is more *computationally efficient*.

The primary assumption undergirding their analysis is that "the computational efficiency of a representation depends on. . .three. . .factors—data structure, program, [and] attention management—and on how well they work together" (ibid., p. 68). Implicit to this assumption is (1) that sentential and diagrammatic representations give rise to differing (external) data structures[3] , and (2) that the productions (i.e. the program) "that operate on [those data structures] are in the problem solvers' [or computer science students'] head" (ibid. p. 67). Based on this assumption, they concluded that "whether a diagram (or any other representation) is worth 10,000 words depends on what productions are available for *searching* the data structure, for *recognizing* relevant information, and for *drawing inferences* from that information" [ibid., p. 68 (emphasis mine)].

Larkin & Simon point out that, in general, the differences in diagrammatic and sentential data structures can profoundly affect the processes of *search*, *recognition*, and *inference* applied by the human program. First, searching a sentential representation's data structure requires a linear search down the data structure to find each informational item. In contrast, since related information is most often grouped together in a diagrammatic representation's data structure, all needed information can often be obtained "for free" by finding the first informational item. Second, "[e]ase of recognition may be strongly affected by what information is explicit in a representation, and what is only implicit" (ibid., p. 70). Unlike sentential representations, "diagrammatic representation[s] [preserve] explicitly the information about the topological and geometric relations among the components of the problem" (ibid., p. 66); consequently, it is often the case that fewer (explicit) inferences need be drawn from diagrammatic representations than from sentential representations, resulting in more efficient computation. Third, and in contrast to the other two processes, "[i]nference is largely independent of representation *if* the information content of the two sets of inference rules is equivalent—i.e., the two sets are isomorphs" (ibid., p. 71). In other words, differing external representations do not lead to more or less powerful inference rules' being applied; to the contrary, the kinds of inferences made depend solely on the inference rules that are available in a person's head.

4.1.3 Discussion

In light of the sample studies presented in the preceding section, it should be clear that much empirical evidence supports the two qualitative statements we proposed in Section 4.1.1: (1) that people remember pictures more easily and readily than text; and (2) that pictures provide a more

[3] The data structure is linear (one-dimensional) in the case of sentential representations, and planar (two-dimensional) in the case of diagrammatic representations.

efficient encoding of concepts than do words. It should also be clear, however, that the empirical evidence in support of those statements has a very general and unapplied flavor; indeed, the studies we examined employed common, everyday pictures and words, and tested subjects in settings far different from the computer science classrooms and labs we examined earlier. Consequently, although the idea that pictures are better than words seems to be generally true, we ought to be wary of applying it to computer science education without more focused empirical evidence.

Spurred by the availability of the computer as a testbed for human processing models, the work of Anderson,[4] Larkin, and Simon we examined above represents a recent trend toward viewing the processing of pictures and text in terms of a computational model. As Larkin & Simon demonstrated, such a model can be extremely useful within the domain of physics and geometry problem solving; indeed, in order to determine the conditions under which diagrammatic representations are superior to sentential representations in solving problems, they merely had to cast problem statements into corresponding sentential and diagrammatic data structures, and then to solve the problems using each alternative representation. The superior representation amounted to the representation that yielded *less search* and *more recognition*—i.e. more matches with the preconditions of a common set of inference rules—to arrive at the solution to the problems. Notice that their analysis depended intimately on the fact that in physics, geometry, and related problem-solving domains, one most frequently uses diagrams to work toward an explicit goal—*viz.*, the solution to a problem. Hence, Larkin & Simon were able to determine that a diagram was better than text if it facilitated a solution to a problem more quickly and easily than an informationally equivalent textual representation.

The same explicit goal cannot be found, however, within the domain of AV. On the contrary, AV's main goal of accelerating and enhancing the process of learning an algorithm is extremely amorphous, and, in general, there exists no litmus test for determining exactly when a student has gained an understanding of an algorithm. We might argue that one has gained an understanding of an algorithm when one passes some appropriate exam, when one is able to integrate that algorithm into one's own computer programs, or when one is able to alter that algorithm subtly to produce an algorithm with different but extremely useful functionality. It seems unlikely, however, that we would be able to agree on any single test for understanding.[5]

Consequently, although Simon & Larkin's work gives us insight into how pictures might be better than text in solving domain-specific problems, it does little to tell us how pictures might be better than text in learning and understanding the dynamic behavior of a computer program. Unless we are willing to define a concrete goal for a visualization session—for example, "Use the visualization to determine what the output of this program will be"—the analysis they used simply does not apply to AV.

In the following two subsections, our focus progressively narrows, as the assumptions that we examine become more specific to AV. As was the case in this subsection, we shall see that the more specifically we can express both our assumptions, and the instructional goals we want AV to meet, the better we will be able to support the assumptions empirically and theoretically.

---

[4] See (Anderson 1982, 1983, 1987, 1989; Anderson, Farell, & Sauers 1984) for a wealth of information on Anderson's ACT* theory, which espouses a production system model for cognitive skill acquisition and performance.

[5] I can imagine at least one situation, however, in which an analysis similar to Larkin & Simon's might be useful in computer science problem solving. It has been my experience that professors like to give exam questions that present a Pascal (or other appropriate programming language) version of an algorithm, and request that students trace the execution of that algorithm given a certain set of inputs. Here, data structure diagrams that kept track of program states might produce the same kinds of search and recognition advantages that Larkin & Simon noted above.

4.2 Graphical software has intrinsic merit

4.2.1 Reformulation

AV practitioners have assumed that because of their sophisticated and aesthetically-pleasing graphics, AV systems naturally lure users to them—that they possess some intrinsic ability to captivate users and to hold their attention. Further, AV practitioners argue that it is precisely because users are naturally captivated by AV systems' graphics that a process of enhanced learning can take place. The belief is that users whose interest naturally focuses on a learning medium will not only learn more quickly, but will absorb more detail, and remember that detail longer. It appears that two key questions are at stake here: (1) whether graphics-based AV systems actually succeed in capturing students' attention; and (2) provided that students are indeed captivated by AV systems, whether their natural attraction to that medium actually facilitates faster and deeper learning than would take place without it (i.e. using conventional pedagogical methods such as textbooks and lectures alone). We examine some theories and research that attempt to address those questions below.

4.2.2 Theories and Empirical Evidence

Unfortunately, to my knowledge, no published studies exist that address the first question directly; hence, we shall have to rely on the informal observations I made during my three years of work in Lawrence's AV lab . The AV practitioners with whom I have collaborated or come in contact[6] seem to agree that AV definitely captures students' attention, so long as students are provided with enough *structure* and *guidance*. Structure should come in the form of laboratory exercises designed to fulfill a definite pedagogical goal—usually related to the concepts being discussed in concurrent lectures. Guidance should come in the form of one laboratory assistant per ten students, who should be available to answer questions, give hints, and get students out of trouble during the duration of a laboratory period. Provided these two ingredients exist, I believe it is safe to answer Question (1) affirmatively, although some sort of empirical study is certainly warranted.

To answer Question (2), we clearly need to concretize the meaning of "faster and deeper learning," or, at the very least, to identify the conditions under which faster and deeper learning is most likely to take place. Salomon (1983) provided an account of two lines of research—one in memory and cognition, the other in social psychology—that have converged to address this issue. Within the area of memory and cognition, the concepts of *maintenance rehearsal* and *elaborative rehearsal* (Salomon 1983, p. 43) have been used to distinguish between, on the one hand, merely repeating material without thinking about it, and on the other, actually relating it and assimilating it to current knowledge. Elaborative rehearsal, it is claimed, "addresses the material at deeper levels and facilitates long term memory of the material" (ibid., p. 43). Similar distinctions in the area of memory and cognition have been made between shallow and deep processing with the terms *involuntary recall* and *deliberate memory*; and, alternatively, with the terms *automatic elaboration* and *controlled and nonautomatic elaboration* (ibid., p. 43).

In a similar vein, the terms *mindlessness* and *mindfulness* have been used within social psychology to differentiate between, on the one hand, "the absence of active conscious information, where the individual relies on the structure of the situations representative of its underlying meaning" (ibid., pp. 43–44), and, on the other hand, "a cognitively active state characterized by conscious

---

[6] Note that in 1991, Professor Tom Naps of Lawrence University received a grant from the National Science Foundation to host a series of two one-week workshops that brought AV practitioners from around the nation together to discuss AV in undergraduate education and to exchange ideas. I served as a laboratory assistant at those workshops, and had the opportunity to discuss this issue with its participants. In addition, Tom Naps and I have discussed this issue at length.

manipulation of the elements of one's environment, in which case the individual questions old categories or constructs new ones" (ibid., p. 44).

As we can see, whether we use the memory and cognition definition or the social psychology definition, the key element of deeper learning is "the effortful, nonautomatic elaboration of the encountered material" (ibid., p. 44). Therefore, the answer to Question (2) lies in determining whether AV possesses the factors required to activate effortful and nonautomatic cognitive processing in learners—that is, whether AV causes learners to expend the mental effort required to learn. Below we examine two theories—one dealing with intrinsic motivation, the other dealing with the learner's perception of the learning medium—that define conditions under which increased mental effort during learning might take place. The discussion that follows in Section 4.2.3 will apply these theories to AV in an attempt to determine whether AV facilitates deeper learning.

Malone's theory of intrinsically motivating instruction. Malone (1981) addressed a factor in learning that he believed to be both "potentially overpowering" and "largely neglected" (p. 334): the role of motivation. Inspired by such eminent learning theorists as Piaget (1951) and Bruner (1962), Malone argued that *intrinsically motivating activities*—i.e., those in which people engage "for [their] own sake, . . . [and not] . . . in order to receive some external reward such as money or status" (Malone 1981, p. 335)—naturally facilitate deep learning. As Malone put it,

> If students are intrinsically motivated to learn something, they may spend more time and effort learning, feel better about what they learn, and use it more in the future. . . . [T]hey may [also] learn better in the sense that more fundamental cognitive structures are modified, including the development of such skills as learning how to learn. (ibid., p. 335)

Having assumed that intrinsically motivating environments positively influence learning, Malone's research attempted to identify the precise features of those environments that might make them intrinsically motivating. Since, at the time of his research, computer games were a relatively new and novel phenomenon, and since "computer games are especially clear illustrations of how the unique capabilities of computers can be used to create motivating environments" (ibid., p. 340), Malone decided to study childrens' opinions on and interaction with computer games in order to gain insight into the particular features of intrinsically motivating environments. While the details of his empirical studies are not important here[7], three general features of intrinsically motivating games emerged—*challenge*, *fantasy*, and *curiosity*; these formed the basis on which his framework for a theory of intrinsically motivating instruction was developed.

Intended to be used as a "checklist of heuristics to be used in designing instruction environments" (ibid., p. 364), Malone's framework provides some precise notions of challenge, fantasy, and curiosity within the context of learning environments.

Challenge. In order for an environment to be challenging, it should provide minimally (1) a set of goals, and (2) an uncertain outcome. However, if the environment is to be used primarily as a *tool* instead of as a *toy* , the criteria change: "Since a good tool is designed to achieve goals that are already present in the external task, it need not provide a goal. Furthermore, since the outcome of the external goal . . . [(e.g. understanding an algorithm in execution)] . . . is already uncertain, the tool itself should be reliable, efficient, and usually invisible" (ibid., p. 359). Finally, self-esteem plays an important role in challenge, since a one's self-esteem can be positively or negatively

---

[7] The studies themselves were nonetheless interesting. In one of his studies, Malone gave 60 grade school children a survey that presented them with a list of the 25 most popular computer games (according to the students' teachers); the students rated each computer game on a four point scale, where 0 meant they had never played the game, 1 meant they did not like the game, 2 meant they liked the game, and 3 meant they liked the game a lot. He used the results of this survey to identify the common features of the best-liked computer games. In two subsequent studies, Malone identified the features of the computer games that made them motivating, and designed multiple experimental versions of two popular games—Breakout and Darts—that allowed him to vary the motivating features "in all sensible combinations" (Malone 1981, p. 345). He had groups of students play different versions of his games, and asked them which versions they liked best. In this way, he could confirm or disconfirm that the hypothesized game features were intrinsically motivating.

influenced according to one's success within a learning environment. It is therefore important not only to reward students with appropriate feedback when they succeed, but also to present failure feedback in such a way that the possibility of self-esteem damage is minimized (ibid., p. 360).

Fantasy. Malone adopts the *American Heritage Dictionary* definition of fantasy ("mental images of things not present to the senses or within the actual experience of the person involved"), and defines three dimensions along which fantasies in intrinsically motivating learning environments can be discussed: (1) *intrinsic* versus *extrinsic* (where *extrinsic* means that the fantasy depends on the use of the skill, and *intrinsic* means that the fantasy not only depends on the skill, but the skill also depends on the fantasy); (2) their *cognitive aspects* (the extent to which they employ metaphors and analogies involving old knowledge to help students acquire new knowledge); and (3) their *emotional aspects* (the extent to which they fulfill the emotional needs of students).

Curiosity. In order to evoke curiosity, a learning environment "should be neither too complicated nor too simple with respect to the learner's existing knowledge. It should be novel and surprising, but not completely incomprehensible" (ibid., p. 362). Malone distinguishes between two types of curiosity—*sensory* and *cognitive*. "Sensory curiosity involves the attention-attracting value of changes in the light, sound, or other sensory stimuli of an environment" (ibid., p. 363). In contrast, cognitive curiosity "can be thought of as a desire to bring better form to one's knowledge structures" (ibid., p. 363), where the desired form is one in which knowledge is *complete*, *consistent*, and *parsimonious*. According to Malone, the way to engage learners' cognitive curiosity is to provide them with " just enough information to make their existing knowledge seem incomplete, inconsistent, or unparisimonious" (ibid., p. 363).

Salomon's Theory on Differential Perceptions and the Investment of Mental Effort. While acknowledging that "the nature of stimuli, their complexity, novelty, structuredness, pace, and the like" (Salomon 1983, p. 45) can all influence learning to some extent, Salomon argues that one's "[p]erceptions, in the sense of predispositions, preconceptions, attitudes, or attributions also play an important role in the way one treats information" (ibid., p. 45). He supports his claim with a study he conducted in which he and his colleagues

> measured, among other things, the way television and print are perceived, how deep or shallow they are in the subjects' eyes, and how easy or difficult they usually are to comprehend. We found systematic and positive relations (r = .32 to .36) between the way a source of information is initially perceived (how deep and demanding it is) and the amount of effort students report investing in a particular subsequent presentation of material from that source (ibid., p.46).

In addition to the depth and mental demand students ascribe to a source of information, Salomon points out two other types of perceptions that affect the amount of mental effort students are likely to expend in learning from a particular medium. First, cultural stereotypes characterizing the substance associated with a particular source can affect mental effort. For example, Salomon found "the material presented on TV is perceived to be shallower and less variable than the material presented in print, even when the content areas (e.g., adventure stories, sports, science) are held constant" (ibid., p. 46). Consequently, even if they comprise the same information, a TV program and an article will evoke different amounts of mental effort in the people watching and reading them.

Second, one's perceived self-efficacy in processing material from a particular source can effect the amount of effort one invests in learning from that source (ibid., p. 47). The relationship between perceived self-efficacy and amount of expended effort is as one might expect: the better one judges one's self efficacy to be at a particular task, the more effort one in likely to expend at that task. This connection, according to Salomon, has been well documented in the literature, including (Bandura, 1978, 1982; Butkowsky & Willows, 1980; Covington & Omelich, 1981; Shavelson, Caldwell, & Izu, 1977).

4.2.3 Discussion

As we can see, Malone believes that intrinsically motivating learning environments naturally give rise to deep learning. Unfortunately for us, his belief rests not on empirical studies, but on his own intuition—although, as he points out, his intuition accords well with the published cognitive theories on the subject. Indeed, that intrinsic motivation facilitates better and deeper learning was not the focus, but rather the starting point of Malone's research, which, as we have seen, purported to unveil the features of instructional environments that might make them intrinsically motivating. Therefore, although it appears we cannot find any empirical proof that intrinsic motivation leads to better learning, if we are willing to take that assumption as a given, then Malone's "checklist of heuristics" can shed light on whether the AV system in question constitutes an intrinsically motivating environment. Let us now scrutinize GAIGS vis-á-vis Malone's three primary criteria for intrinsically motivating environments.

First, is GAIGS *challenging*, i.e., does it provide a set of goals and an uncertain outcome? Since GAIGS is definitely a tool and not a toy, its goal lies in the external task of AV: to elucidate the logic of a computer algorithm. As for providing an uncertain outcome, GAIGS serves to reveal the effect of an algorithm on its key data structures as the algorithm unfolds; insofar as GAIGS is used to gain an understanding of an algorithm whose details are not yet understood, the outcome of that algorithm, as manifested in the algorithm's actions, is uncertain.

Second, does GAIGS provide *fantasy*, i.e., does it contain "mental images of things not present to the senses or within the actual experience of the person involved?" While the data structure pictures central to GAIGS's algorithm visualizations should be meaningful to viewers—indeed, they are consistent with the data structure figures found in most computer science textbooks—they nonetheless constitute abstract representations of collections of 0's and 1's in a computer. Clearly, insofar as these abstractions are intended to conjure up mental images of things not present (0's and 1's in the computer) to the senses of viewers, they constitute a fantasy as Malone defines it.

Third, does GAIGS evoke curiosity? That is, is it "neither too complicated nor too simple with respect to the learner's existing knowledge" . . . [, and is it] . . . "novel and surprising" (Malone 1981, p. 362)? Whether GAIGS meets the criterion of curiosity depends intimately not only on the knowledge of its users, but on the the precise tasks those users are assigned in GAIGS. Since GAIGS serves as an extremely flexible pedagogical tool that may be used to present a nearly infinite number of algorithms in many different ways, it follows that for every student, there exists an exercise in GAIGS with a sufficient level of complexity, novelty, and surprise to evoke curiosity. Clearly, the burden of curiosity rests not on GAIGS, but on the educators who use it as an instructional tool; it is up to them to construct exercises in GAIGS that are sufficiently novel and surprising—and, at the same time, embody a sufficient amount of complexity—to elicit curiosity in their students. *Prima facie*, it appears that GAIGS constitutes a Malonian intrinsically motivating environment.

Moving to an analysis of GAIGS with respect to Salomon's work, we find that empirical studies are needed to measure students' perceptions of AV as a learning medium before we can analyze GAIGS with respect to the pool of research that Salomon illuminates. Such research would most likely come in the form of surveys; students studying algorithms in a class that had access to GAIGS could be asked to compare reading about an algorithm with visualizing that algorithm in GAIGS. Such questions as "Which medium do you perceive to be more difficult to comprehend?"; "Which medium gives you a better idea of how the algorithm works?"; and "Which medium do you think lends itself better to the way you think about the algorithm?" could shed light on students' perceptions of (1) the mental effort each medium requires; (2) each medium's depth or substance, and (3) their self-efficacy with each medium. Students could be asked to answer questions on a five point scale (on which, for example, 0 would be neutral, negative numbers would favor reading, and positive numbers would favor AV), and the responses could be statistically analyzed to determine trends in

students' perceptions of each source. These trends, according to Salomon, would give us an indication of the amount of mental effort students are willing to expend on each source, and hence the relative number of effortful, nonautomatic elaborations—which, as the theory espouses, give rise to deeper learning— that each medium evokes.

Although an empirical study is clearly needed to identify precisely students' perceptions of AV, our informal observations in the Lawrence AV lab have given us an intuitive feel for how students process the information conveyed in an algorithm visualization. Our experience suggests that when AV reverts to a passive activity—i.e., when students are not given a laboratory goal and a structured way to meet that goal—students tend to treat AV more as an entertainment device than as a learning device. Captivated by the aesthetically pleasing appearance of the data structure pictures, students seem to view those pictures more as a form of artwork, and less as an abstraction meant to provide insight into the dynamic behavior of an algorithm. On the other hand, when students are given structured laboratory exercises with a specific and clearly stated instructional goal[8], they seem to focus more intently on the visualization sequence—presumably, so that they can gather the knowledge required to complete the exercise. Moreover, the presence of lab assistants to oversee student activity, and to intervene when necessary, also seems to help students focus on the visualization[9]. In short, our casual observations suggest that AV practitioners can promote the kind of deeper cognitive processing that enhances learning simply by (1) providing students with structured laboratory activities; and (2) providing knowledgeable lab assistants to oversee and assist students in the lab.

4.3 Algorithm visualization is more effective than conventional methods in teaching algorithms; it accelerates and enhances the learning process

4.3.1 Reformulation

This final assumption is the most AV-specific of the ones we examine in this paper; indeed, it lies at the heart of AV's advancement as an instructional tool. And in our examination of the theories and research that respond to the two assumptions above, we have already caught a glimpse of how cognitive science might respond to it. For example, in Section 4.1 we learned that, generally speaking, pictures are more effective than words in encoding information, and that they have been shown to facilitate faster recall and longer memory retention than text. Furthermore, in Section 4.2 we saw that visual computer-based instructional tools like GAIGS constitute Malonian intrinsically motivating environments, which, as has been postulated, promote the controlled and nonautomatic cognitive elaborations necessary for deep and substantial learning to take place.

Although all are applicable to AV, none of the studies we have examined thus far addresses its pedagogical impact specifically. Any empirical response to the assumption at hand, however, must be very specific; ideally, it would compare students' interaction with some specific AV system to their interaction with more conventional teaching media like textbooks, lectures, and articles. After nearly six months of unsuccessful literature searches, I came to the conclusion last November that if I really wanted empirical evidence on the specific effects of AV in learning algorithms, I would have to conduct the studies myself.

---

[8] See Figure 3, the Shell Sort lab, for an example.

[9] Interestingly, this observation accords well with a study on childrens' TV viewing behavior (Salomon, 1977). Salomon found that when mothers co-observed educational TV programs with their children, the children experienced "large gains in knowledge and skill acquisition" (Salomon 1983, p. 48) as compared to when they viewed the programs alone. Similarly, it seems that when students work under the supervision and guidance of a lab assistant, they concentrate more on the visualizations, which may lead to the same kinds of large gains in knowledge and skill acquisition that Salomon observed.

Much to my delight, however, I discovered in late November, 1992 that several prominent researchers had beat me to it.[10] Indeed, John Stasko, Albert Badre, and Clayton Lewis will be presenting a paper entitled "Do Algorithm Animations Assist Learning? An Empirical Study and Analysis" at the 1993 ACM SIGCHI (Special Interest Group on Computer-Human Interaction) Meetings in April. In Section 4.3.2 below, we examine this study. To my knowledge, it is the only one of its kind, and the only one that specifically addresses the assumption that AV is more effective than conventional teaching methods.[11]

4.3.2 Theories and Empirical Research: Stasko, Badre, & Lewis Study

Background. (Stasko, Badre, & Lewis, 1992) focused on students' use of TANGO (Transition-based ANimation GeneratiOn) (Stasko 1990), an algorithm animation system Stasko developed as part of his dissertation work at Brown University in the late 1980s. Like GAIGS, TANGO is driven by a mouse-based graphical interface, which allows users to interact extensively with a visualization. In contrast to GAIGS, however, TANGO supports *visualization through animation* as opposed to GAIGS's *visualization through pictures*; whereas GAIGS depicts an algorithm in execution as a series of static data structure pictures, a visualization in TANGO progresses as an animated film, with smooth transitions between dynamic program states.

For example, consider the insertion sort algorithm that we explored earlier. In one possible animation of that sort in TANGO, data elements are represented as sticks whose height corresponds to magnitude. The two data elements that are currently being compared are highlighted, and when a stick is moved within the array it actually bounces from its current position to its new position. As the sort progresses, users view the line of sticks as they smoothly and continuously fall into order. When the sort is finished, the tops of the sticks form an ascending (or descending, depending on the type of sort) ramp, graphically indicating that the data are in order.

Method. Stasko and his colleagues at the Georgia Institute of Technology gathered twenty subjects—all volunteers and graduate students. The subjects were evenly divided into two groups: one that would have, in addition to a couple journal articles, the benefit of TANGO in learning the *pairing heap* implementation of a *priority queue*,[12] and another that would have to rely on the journal articles alone. The control group was given 45 minutes to read about and study the pairing heap articles, whereas the animation group received 30 minutes to read about pairing heaps, and an additional 45 minutes to explore it in TANGO using an interactive animation.[13] At the end of each respective learning session, subjects were given a written exam consisting of 24 questions intended to test several facets of the pairing heap algorithm. In order to do well on the exam, subjects needed to possess minimally (a) declarative or factual knowledge of the algorithm; (b) an understanding of the algorithm's computational complexity; and (c) procedural knowledge of the algorithm. The researchers hypothesized that the algorithm animations "would be more beneficial to procedural understanding than to declarative understanding" (Stasko, Badre, & Lewis 1992, p. 3). This hypothesis stemmed from their observation that although algorithm animations "depict how an algorithm functions,. . .they do not explain the reasoning for particular actions" (ibid., p. 3).

---

[10] Bear in mind that I first wrote this paper in late 1992 and 1993.

[11] Since the original release of this paper in January of 1993, more comprehensive empirical studies have been conducted. Indeed, Andrea Lawrence completed a dissertation at the Georgia Institute of Technology that specifically addresses the empirical evaluation of algorithm animation in educational settings. See (Lawrence, Badre, & Stasko, 1994).

[12] Note that none of the subjects had previously studied the pairing heap, which is a relatively new implementation for the priority queue., a FIFO (first in, first out) data structure in which entries may be ordered according to their priority number: entries of higher priority leave the queue before those of lower priority.

[13] The animation was interactive in the sense that it allowed subjects to choose an operation to perform, and then graphically depicted the effect of that operation on a graphical representation of the pairing heap.

Results. Figure 5 compares the scores of the text-and-animation group to those of the text-only group. As the scores suggest, the main and rather disappointing result of the study was that algorithm animations seemed not to aid students significantly in their understanding of the pairing heap algorithm. To the contrary, "while animation subjects outperformed text-only subjects, the difference was not large and was not statistically significant"[14] (ibid., p. 4). Moreover, "the [exam] performance of the animation group was not high in absolute terms" (ibid., p. 3), with an average score of only 13.6 out of a possible 24.

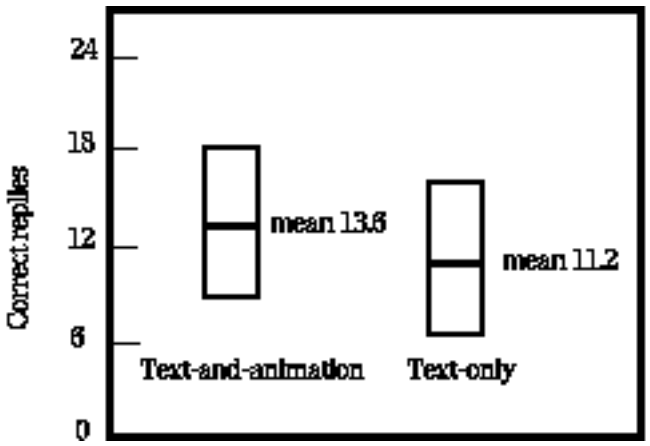

Figure 5. Comparison of test scores of text-and-animation and text-only groups (adapted from (Stasko, Badre, & Lewis 1992, p. 4)

If we break scores down according to question type, we find that the animation group scored consistently better (albeit by a margin that was deemed statistically insignificant) than the text-only group on the declarative and analytical questions; however, that margin of difference narrows considerably if we consider the groups' scores on the procedural questions. Notice that this observation supports the researchers' hypothesis that algorithm animations would not aid procedural understanding as much as they would aid declarative understanding.

Stasko *et al.* administered a post-exam questionnaire to the animation group in order to gather some subjective, introspective data on the effects of the animation. They found that all members in the animation group believed the animation assisted in their understanding of the algorithm. Four reasons were cited. First, and most important, "the animation grabbed their interest" (ibid., p. 4). Second, they liked the interactivity of TANGO; they appreciated being able to explore their own examples. Third, they found the visual effect of highlighting to be valuable in shifting their focus to the crucial portion of the animation. For example, when two nodes in the pairing heap were being compared, each node changed to a brighter color. And finally, the animation subjects appreciated the smooth transitions among animation states. As one subject put it, "I liked how the computer slowly changed the diagram, letting me see how the change was made rather than an instantaneous change" (ibid., p. 4).

On the other hand, subjects criticized the animations on four fronts. First, they longed for concomitant textual descriptions—or perhaps voice annotations—to reinforce what was going on during each stage of the animation. Second, they wanted the ability to step through the animation a frame at a time. Third, they wanted the ability to rewind the animation at any point and play it

[14] The authors performed a two sample t-test on the exam scores and found "a nonsignificant trend favoring the animation group ($t=1.111$, $df=18$, $p < 0.13$)" (Stasko, Badre, & Lewis 1992, p. 3).

back. And finally, several subjects noted that although they understood was happening during an animation, they found it difficult to remember afterwards.

4.3.3 Discussion

In interpreting the study's rather disappointing results, we should first look for possible logistical problems with the method the study employed. Stasko and his colleagues were successful in identifying two of these. First, one of the researchers (Clayton Lewis) discovered a logical explanation for the ostensibly low absolute scores. By way of a programming walkthrough, he determined the knowledge that users could acquire from the readings and from animation viewing. When he compared the acquirable knowledge to the knowledge required to score well on the test, he discovered that "neither the text nor the animation group could be expected to produce high performance" (ibid., p. 4). Second, the researchers surmised that both groups might have benefited from knowing what types of exam questions to expect; this information was not revealed to either group before the tests, which meant that subjects could not cater their learning in a way that would prepare them for the test. To these two problems, I would add a third: insofar as the subjects' academic background would have influenced their ability to interpret the animations meaningfully, the subjects' past experience with the TANGO algorithm animation system in particular, and with AV in general, ought also to have been considered in analyzing the final results. I believe that those subjects with more previous exposure to AV might have already gained a feel for how to process animations—and hence might have been in a better position to obtain the knowledge necessary to well on the exam—than those subjects with no previous exposure to AV.

Having identified some problems with and sources of error in the study, we are in a position to analyze the results and draw some conclusions. It should be clear that even though the results did not overwhelmingly support the hypothesis that AV assists learning, they did shed light on the fundamental problems with AV in its current form, and on how AV might be used more effectively. In particular, at least six general conclusions salient to the focus of this paper may be drawn from this study.

The first (and, in my opinion, the most intriguing) conclusion of this study might seem rather obvious on the surface: "For a student to benefit from the animation, the student must understand both [the mapping from algorithm abstractions to graphics], and the underlying algorithm upon which the mapping is based" (ibid., p. 4). Since that mapping will probably not be understood by novices who are first learning an algorithm, it appears that AV will not benefit novices as much as it will benefit experts who are studying key behaviors of, or subtle variations on, algorithms that they already know and understand. The authors hypothesize that a more beneficial activity for novices might be to construct animations themselves (ibid., p. 5), since such a construction would serve to elucidate the algorithm abstraction to computer graphics mapping that they most likely do not yet understand.

Second, the study revealed that since students do not always understand the semantics of a visualization sequence, it would be helpful if textual or audio explanations were available on-line to reinforce what is going on. Hence, it appears that some mixture of text, graphics, and sound may be more useful in learning algorithms than text, graphics, or sound alone.

Third, although an animation may convey the essential steps of an algorithm, it fails to present the motivation underlying those steps. Hence, complementary motivational instruction—whether through pre-lab lectures, on-line help, or in-lab assistants—will enhance students' understanding of an algorithm, and allow them to interpret a visualization more meaningfully.

Fourth, the authors' original failure to consider the precise knowledge that subjects would need in order to do well on the exam points out the importance of gearing a visualization exercise to specific

instructional goals. As the authors admit, "there was no clear reason to think that viewing the animation would help learners perform the tasks we asked them to perform" (ibid., p. 6). Notice that this conclusion accords well with the informal observations we presented in Section 4.2.3 above, in which we underscored the need for labs to have specific goals, along with a structured plan for meeting those goals.

Fifth, contrary to the research we examined in Section 4.1, which showed that people are generally able to remember pictures longer than words, Stasko *et. al.*'s study suggests that in the realm of algorithm understanding, the opposite may be true: students may retain information about algorithms that they read longer than information that they see in a visualization. Indeed, as we saw above, although understanding it at the time, several students reported difficulties in remembering an animation after it finished. Interestingly, this observation is supported empirically by a related study (Palmiteer & Elkerton, 1991) that compared how well two people could learn from two different sources—animated demonstrations and textual descriptions. The study found that although the animation group initially perfomed the tasks more quickly and accurately than the text-only group, after seven days they did not retain what they had learned as well as the text-only group.

Finally,the results of the post-exam questionnaire support our conclusion in Section 4.2 that algorithm visualization systems constitute intrinsically motivating environments. Not only did students report that "the animation grabbed their interest" (ibid., p. 4), but the authors saw students' natural attraction to it as "one of the primary advantages of algorithm animation" (ibid., p. 4).

5 Conclusion

So we have seen that the three primary assumptions, on the basis of which AV has been recently advanced, require further refinement and qualification in order to find corroboration within the theories and empirical studies of cognitive science. The first assumption—that pictures are better than words—has a well-established empirical basis in general cognitive psychology; however, as we found, the evidence in support of the assumption is not sufficiently focused to allow us to conclude that it is necessarily valid within the domain of AV. In fact, the AV-specific research that we examined in Section 4.3.2 (Stasko, Badre, & Lewis, 1992; Palmiteer & Elkerton, 1991) suggested that, within the context of AV, the contrary may be the case.

The second assumption—that AV graphics-based environments motivate learning—seems to secure some theoretical support within cognitive science, so long as we are willing to acknowledge, without substantial empirical evidence, the existence of two necessary preconditions : (1) that AV environments capture students attention, i.e., that they constitute intrinsically motivating environments; and (2) that intrinsically motivating environments naturally give rise to deep learning. Apropos of (1), we saw that many computer science educators agree that at least two provisions can improve the likelihood that AV environments will be intrinsically motivating: (a) All AV sessions should have explicit structure and goals; and (b) students interacting with AV systems should have access to extensive on-line or human assistance to help them when problems or questions arise. Apropos of (2), we found that researchers have either taken this fact for granted (Malone, 1981), or have attempted to identify specific conditions under which deeper learning might take place (Salomon, 1983). Unfortunately, due to the lack of empirical studies that specifically address, within the domain of AV, the motivational and perceptual issues raised by the researchers above, the most we can say about AV's ability to captivate students and facilitate deeper learning is

that there seem to be certain things educators can do to create AV learning environments that promote deep learning better than others.[15]

Finally, in support of the third assumption—that AV is more effective than conventional methods in teaching algorithms—we found only one isolated empirical study; and, as we saw, its results were far from unequivocal. Nonetheless, it did provide significant insight both into AV's strengths and weaknesses as a pedagogical tool, and into how we might make AV more effective. The conclusions we drew from this study point to some interesting directions in which future research in AV ought to progress. We conclude below by describing two of the most exciting of these.

Finding alternative methods for evaluating AV empirically. Two main obstacles stand in the way of researchers interested in conducting studies to analyze AV empirically.

First, since computer-based instruction in general, and AV in particular, are relatively new pedagogical devices, we have not yet been able to agree on methods for empirically studying them. Although the logical approach might be the one Stasko, Badre, & Lewis (1992) took—to assemble students into text-and-visualization and text-only groups, and then to administer a post test and questionnaire—we saw the difficulties with such an approach. For example, the extremely closed conditions under which the study was conducted, and the short period of time it spanned, deviate significantly from the conditions under which AV is normally used.[16] Indeed, as we learned, AV is typically used on numerous occasions and over a period of many months. Furthermore, it was not at all clear that the written exam the researchers administered gave them any indication of the students' understanding of the algorithm—partly because the whole concept of understanding is so amorphous, and partly because the exam, as they found out later, asked questions that they could never have expected the students to answer.

Second, as Stasko, Badre, & Lewis accurately point out,

> it is [pragmatically] challenging to assemble the appropriate ingredients for an empirical study. An algorithm animation environment must be available, and most importantly, a group of subjects who are at an appropriate point in their educational careers must be available. Even this may not be enough, however, because splitting the subjects into two groups, one using animation and one not using animation, may unfairly influence student achievement and grading in the particular course in which the students are enrolled. (ibid., p. 1)

Consequently, we see that researchers may never be able or willing to construct the setting that would be most appropriate for an empirical study.

Clearly, on this front, the primary challenge facing future researchers lies in developing alternative methods for empirically analyzing AV's effectiveness as a learning tool. One promising alternative, an ethnographic method called *conversational analysis* (CA) (Suchman, 1987), constitutes a radical departure from the empirical method Stasko *et. al.* used. CA entails videotaping and analyzing two-person teams as they interact with an AV system to complete an appropriate series of laboratory exercises. It is argued (Douglas, 1992) that CA can be extremely effective in pinpointing user misconceptions in their interaction with software. Since two subjects must work together as a team, they must make explicit to each other their understanding and misunderstanding both of the software with which they are interacting, and of the concepts they are studying in that interaction. Since all interaction is captured on videotape, researchers can thoroughly analyze it— both to identify user misconceptions and to determine the reasons for those misconceptions.

At least two advantages to CA are readily apparent: (1) it can be carried out under conditions that are far more easily attainable than those required in an empirical study like Stasko *et. al.*'s; and (2)

---

[15] And, of course, that AV-specific studies addressing these issues are certainly warranted.

[16] See Section 2.3.3.

it has a proven track record[17] of successfully identifying user misconceptualizations, which, in turn, has allowed researchers to improve their software systems.

Developing environments that facilitate greater experimentation with alternative abstraction-to-graphics mappings. In all existing AV systems of which I am aware, students can visualize an algorithm only through some limited set of predefined views (i.e. algorithm abstraction-to-graphics mappings); the construction of new views, if possible at all, requires much time and effort. In the case of GAIGS, for example, the set of views is restricted to pictures of nine conventional data structures found ubiquitously in computer algorithms—especially those with which first and second year college students are familiar. GAIGS users are not even provided with instructions for constructing alternative visualizations, because the task of extending algorithm views in GAIGS is far too complicated for anyone but an advanced programmer to tackle.

We see that a serious problem with existing AV systems is that they do not support user experimentation with alternative algorithm views. Even if an AV system user comes up with an idea for some alternative way of viewing an algorithm—one that might better capture the essence of the algorithm—the tremendous time and effort required to build that new, alternative visualization often deters the user from doing so. As a result, insofar as they discourage user experimentation, existing AV systems impair progress that could be made in finding more meaningful visualizations.

We conjecture that the future in visualization lies in systems that enable even novice users to construct their own visualziations quickly and easily. Such systems would be advantageous for at least two reasons. First, they would foster the kind of visualization experimentation that is not presently possible, and that could lead to the discovery of more effective visualizations. Second, they would allow us to explore the educational benefits of constructing a visualization, and they would give us a testbed on which to compare the process of constructing a visualization with the process of viewing one. As Stasko, Badre, & Lewis point out, the process of constructing a visualization may be more valuable than actually viewing one, because it makes the abstraction-to-graphics mapping—which, as we learned in Section 4.3, is paramount to understanding the visualization—explicit.

[17] See (Douglas, 1992).

Bibliography

Anderson, J.R. (1978). Arguments concerning representations for mental imagery. *Psychological Review 85*, 249–277.

Anderson, J.R. (1982). Acquisition of Cognitive Skill. *Psychological Review 89*, 369–406.

Anderson, J.R. (1983). *The architecture of cognition*. Cambridge, MA: Harvard University Press.

Anderson, J.R., Farrell, R., & Sauers, R. (1984). Learning to program in LISP. *Cognitive Science 8*, 87–129.

Anderson, J. R. (1987). Skill acquisition: Compilation of weak method problem solutions. *Psychological Review 94*, 192–210.

Anderson, J.R. (1989). Skill acquisition and the LISP tutor. *Cognitive Science 13*, 467–505.

Bandura, A.D. (1978). Self efficacy: Toward a unifying theory of behavioral change. *Psychological Review 84*, 191–215.

Bandura, A.D. (1982). Self efficacy mechanism in human agency. *American Psychologist 37*, 122–147.

Brown, M. (1988). *Algorithm animation*. Cambridge, MA: The MIT Press.

Butkowsky, I.S., & Willows, D.M. (1980). Cognitive motivational characteristics of children varying in reading ability: evidence for learned helplessness in poor readers. *Journal of Educational Psychology 72*, 408–422.

Carey, S. (1986). Cognitive science and science education. *American Psychologist 41*, 1123–1130.

Cooper, D, & Clancy, M. (1985). *Oh! Pascal!* New York: W.W. Norton & Company.

Dann, W.P. (1990). *Cognitive aspects of programming, programming paradigms, and programming instruction*. Unpublished masters thesis, State University of New York Institute of Technology, Utica/Rome.

Covington, M.V., & Omelich, C.L. (1981). As failure mounts: Affective and cognitive consequences of ability demotion in the classroom. *Journal of Educational Psychology 73*, 796–808.

Davies, S.P. (1991). The role of notation and knowledge representation in the determination of programming strategy: a framework for integrating models of programming behavior. *Cognitive Science 15*, 547–572.

Douglas, S.A. (1992). Using conversational analysis to discover breakdown in the design of user interfaces. In *Proc. 1992 CSCW Workshop on Ethnographic Methods and Design*.

Elsom-Cook, M. (1989). Acquisition of computing skills (pp. 137–166). In Ann M. Colley & John R. Beech (eds.), *Acquisition and performance of cognitive skills*. New York: John Wiley & Sons.

Futernick, K.D. (1988). *A critical examination of computer programming as educative experience.* Unpublished doctoral dissertation, University of California, Berkeley.

Hundhausen, C.D. (1991). *LUAVDE: Viewing real-time algorithm visualization as a physical model.* Unpublished senior independent study thesis, Lawrence University, Appleton, WI.

Katz, I.R. (1988). *Transfer of knowledge in programming.* Unpublished doctoral dissertation, Carnegie Mellon University, Pittsburgh, PA.

Larkin, J.H., & Simon, H.A. (1987). Why a diagram is (sometimes) worth ten thousand words. *Cognitive Science 11*, 65–99.

Lawrence, A.W., Badre, A.N., & Stasko, J.T. Empirically evaluating the use of animations to teach algorithms. Technical report GIT-GVU-94-07, 1994.

Madigan, Stephen (1983). Picture memory. In John C. Yuille (ed.), *Imagery, memory, and cognition: essays in honor of Allan Pavio.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Mayer, R.E. (1975). Different problem-solving competencies established in learning computer programming with and without meaningful models. *Journal of Educational Psychology 67,* 725–734,

Mayer, R.E. (1987). Cognitive aspects of learning and using a programming language (pp. 61– 79). In John M. Carroll (ed.), *Interfacing thought: cognitive aspects of human-computer interaction.* Cambridge, MA: The MIT Press.

Naps, T.L. & Hundhausen, C.D. (1991). The evolution of an algorithm visualization system (pp. 259–266). *Proceedings of the 24th Annual Small College Computing Symposium* , Morris, MN.

Palmiteer, S., & Elkerton, J. (1991). An evaluation of animated demonstrations for learning computer-based tasks (pp. 257–263). In *Proceedings of the ACM SIGCHI '91 Conference on Human Factors in Computing Systems*, New Orleans, LA.

Paivio, A. (1975). Coding distinctions and repetition effects in memory. In G.H. Bower (ed.), *The psychology of learning and motivation.* Vol. 9. New York: Academic Press.

Paivio, A. (1983). The Empirical Case for Dual Coding. In John C. Yuille (ed.), *Imagery, memory, and cognition: essays in honor of Allan Pavio.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Paivio A., & Csapo, K. (1973). Picture superiority in free recall: Imagery or dual coding? *Cognitive Psychology 5*, 176–206.

Rieber, L.P. (1990). Using computer animated graphics in science instruction with children. *Journal of Educational Psychology 82*, 135–140.

Singley, M.K. & Anderson, J.S. (1989) *The transfer of cognitive skill.* Cambridge, MA: Harvard University Press.

Shavelson, R., Caldwell, J., & Izu, T. (1977). Teacher's sensitivity to the reliability of information in making pedagogical decisions. *American Educational Research Journal 14,* 83–97.

Stasko, J. (1990). TANGO: A framework and system for algorithm animation. *Computer 23*, 27–39.

Stasko, J., Badre, A., & Lewis, C. (1993). Do algorithm animations assist learning? An empirical study and analysis. *Procedings of the 1993 ACM SIGCHI Conference* (Amsterdam, The Netherlands), 61–66.

Suchman, L. (1987). *Plans and situated actions: The problem of human machine communication.* Cambridge, MA: Cambridge University Press.

Turing, A.M. (1950). Computing machinery and intelligence. *Mind 59*, 433–460.

Weaver, J.L. (1989). *The complexity of basic computer programming for novices.* Unpublished doctoral dissertation, University of Texas, Austin.